Optical Character Recognition using Deep Learning: An enhanced Approach

Marwa Amara [1]† and Radhia Zaghdoud [2†],,
marwa.amara@nbu.edu.sa[1]† radhia.zaghdouud@nbu.edu.sa[2†],
Northern Border University, Arar 91431, Kingdom of Saudi Arabia[1,2]
Faculty of sciences Arar, Department of Computer Sciences[1,2]
LARIA, National School of Computer Science[1,2]
Manouba University, Tunis, Tunisia[1,2]

Summary
Optical character recognition (OCR) is a critical activity that has a wide range of applications. The most important stage in any system is still the segmentation of the script to be transmitted for the recognition system. Modeling and feature extraction are done with these smaller components that make up the word. As a result, the segmentation stage is thought to be the primary source of recognition mistakes. One of the most serious issues with Arabic character segmentation systems is the absence of research on Arabic character features. In reality, a successful segmentation system necessitates awareness of Arabic topography.
To address this problem, we've created multiple roles that contribute to promote the issue. Then, to decide whether to affirm segmentation points or re-do the segmentation, we suggest a binary support vector machine (SVM). Finally, deep learning is used to classify the characters.

Keywords:
OCR; handwritten segmentation; SVM; deep learning

1. Introduction

The Arabic is spoken by approximately 250 million people and written by more than 100 million people in more than 20 different countries. A variation of style of writing is remarkable between the Mashreq and the Maghreb. Vertical ligatures are more common in Scripting in the Middle East, and writing styles are more diverse. The Arab topography is the most significant impediment to OCR systems' performance. [1]

Many research papers have been published in the area of segmentation and recognition of handwritten Arabic script [2], but these efforts have not yielded satisfying results due to a lack of studies on the topic and the fact that it is more complex than Latin script. Arabic employs a variety of external objects, including '**dots**,' '**Hamza**,' '**Madda**,' and

'**Diacritic**'. Line separation and segmentation scripts become more challenging as a result of these factors.

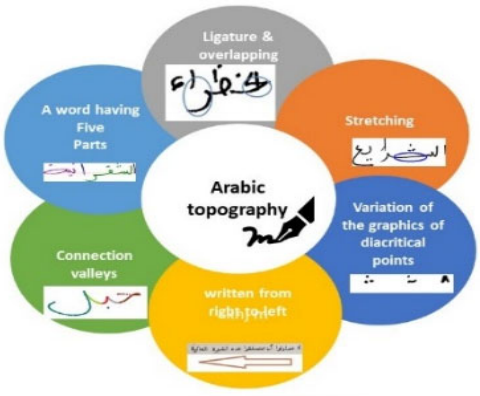


Fig. 1. illustration of several script rules

Depending on their position, Arabic characters can have multiple shapes: beginning, middle, final, or isolated. Many ligatures are used in Arabic, especially in handwritten text as in Fig. 1. Other characters have very similar contours and are difficult to segment and distinguish, especially when there are noise in the scanned image. As a result, the difficult structure of the Arabic language can lead to a failed analysis and be an impediment to achieving a higher recognition rate.

Despite the fact that the research yields interesting results in the field of text recognition, the rates shown by OCR systems trade are quite low [3]. Furthermore, the majority of work on Arabic recognition has been on individual characters, digits, or a small number of words [2].

The remaining of the paper is organized as follows: In Section 2, we provide related works on Arabic text segmentation. Section 3 presents the various modeling methods for Arabic text recognition. Section 4 presents the proposed deep learning algorithm. Conclusions and plans for future work are presented in the concluding part.

Manuscript received May 5, 2022
Manuscript revised May 20, 2022
https://doi.org/10.22937/IJCSNS.2022.22.5.75

2. Character segmentation algorithms

We'll go over the many algorithms that researchers have used to solve the challenge of handwriting segmentation in this section.

2.1 Segmentation based on histogram and baseline

In the literature, there have been numerous attempts to locate segmentation points by evaluating the vertical projection profile.

The study [4] presents a new algorithm for segmenting handwritten Arabic text. The proposed method involves over-segmenting the word and then removing extra breakpoints based on letter shape knowledge. On a test dataset of 200 images, a 92.3% correct segmentation rate was observed, with 5.1% cases of over-segmentation.

[5] proposes a segmentation technique based on a modified vertical histogram generated by deleting dots, ascenders, and descenders, as well as applying a thinning operation. The distance between the top and bottom foreground pixels for each column is used to calculate the modified vertical histogram. The histogram's minima present possible segmentation points. The appropriate segmentation points are chosen based on a distance analysis between prospective segmentation points. The segmentation point validation technique was an important aspect of the segmentation process in this study. It was created with the goal of removing incorrect segmentation points while retaining valid ones in order to improve overall segmentation accuracy. On a 500-word local database, the character recognition rates were 82.98 %.

The approach presented in [6] starts by tracing and straightening the input text-line image's baseline. Subsequently, utilizing features acquired from histogram analysis, it over-segments each word and then removes extra segmentation points using some baseline and language dependent methods. With two independent datasets, researchers tested the proposed algorithm. 93.49 % of the characters in a test set of 200 words from the FN/ENIT dataset were correctly segmented.

The task of segmenting offline handwritten Arabic text up to character level is taken up by Ghaleb et al [7]. To begin, connected components of the word image are extracted using graph-theoretic modeling. These elements are thoroughly examined in order to aid the segmentation of the input image into sub-words. The diacritics are eliminated in the sequel. Then, using two algorithms that use stroke width as a heuristic, a large number of possible segmentation locations are selected. A set of rules on the candidate segmentation points is used to produce the final segmentation points. Finally, each sub-word is segmented, and diacritics are restored to their original positions, taking into consideration diacritics displacement. Experiments are carried out on a set of IFN/ENIT dataset. A correct segmentation rate of 70% was recorded.

The histogram projection technique based on segmentation is quick and easy to use. Cursive text, on the other hand, produces extremely poor results due to the reduced spacing between characters and the shift or skewness of the baseline [8] [3].

2.2 Segmentation based on contour tracing

Romeo-Pakker et al. [9] suggested Two character segmentation approaches. The lowercase writing area in lines is detected using traditional horizontal and vertical projections. A contour-following algorithm that starts in the lowercase writing area and labels the discovered contours solves the problem of overlapping lower or higher strokes. The junction segments connecting the characters are discovered in the first technique by taking the writing line thickness into consideration. The upper contour of each word is detected using the second method. To discover major segmentation locations, the strokes are identified. These points are examined using an automata that analyzes the word's shape while determining decisive segmentation points.

Proposed method has been tested on Arabic and Latin handwritten words dataset containing 1383 words written by 5 different writers. The first approach present a segmentation rate of 93.5 % and the second one achieved the rate of 99.3 % .

Sari et al. [10] employed the contour presentation to detect segmentation locations by applying rules to the local minima of each sub-bottom word's contour. Characters that were vertically overlapped due to the writing style or slant were advanced processed. On a small dataset of 100 words omni authors, the segmentation rate was 86%.

The upper profile minima leads to cut in valleys of letters in contour analysis segmentation, separating the upper short line and dissecting vertical strokes from the body of a character. As a result, it is clear that segmentation based on contour must be combined with other features in order to obtain better rates.

2.3 Other Segmentation techniques

Thinning has been utilized in the recognition of handwritten Arabic characters by some researchers in [11] and [12]. It could be a crucial strategy for resolving the character segmentation issue. Combining it with additional techniques, on the other hand, would be necessary to provide better segmentation. [3]

In the literature, there is very little research on adopting neuronal networks (NN) to segment Arabic characters. Authors in [13] was applied a recursive conventional

method. As a first step, the algorithm produces pre-segmentation points. Next, a NN is employed to check the correctness of these segmentation points. The reported segmentation rate was 69.72%. As a result, NN is a powerful tool, and its use to segment characters is a very promising field for further research[3].

Some researchers attempted to improve the segmentation process by employing morphological operators that had not previously been widely examined for character segmentation[14].

Despite the large number of algorithms that have been proposed, there has been very little research into handwritten characters segmentation based on knowledge. Investigated knowledge about Arabic characters shapes is important to get better results.

As a result, we present a two-step segmentation approach in this paper. The first phase is based on various rules describing the specific characteristics of the Arabic characters shape as well as the vertical histogram of each sub word. Then, we investigate a binary SVM decision to confirm the segmentation points are correct or they need correction.

The original contributions of this work are:

- ✓ Enhance the roles presented in our work [17] to segment printed word in order to segment handwritten. Two new roles was proposed to enhance segmentation accuracy of Arabic handwritten characters. This method lead to reducing segmentation errors.
- ✓ An SVM-based segmentation point validation system is introduced. It verifies the validity of the segmentation points detected by the initial algorithm.
- ✓ Using conventional neural networks (CNNs) to classify segments and report the overall recognition rate.

3. Character segmentation algorithm

In this section, we discuss our segmentation algorithm which is based on the special characteristics of Arabic script. technique takes as input an Arabic word. It generates, at the end of the whole process, a word segmented into characters. Our approach consists of three main steps: word segmentation (WS), special lines detection (SLD) and the preliminary segmentation point detection (PSPD). These stages are depicted in this section.

3.1 Sub-words separation

The initial step in our system is called segmentation (WS). Sub-word detection is achieved by vertically projecting a word's image onto a horizontal axis. There will be some zero value columns in the vertical histogram (Histv). As indicated in the figure 2, these columns are utilized to separate the connected parts.

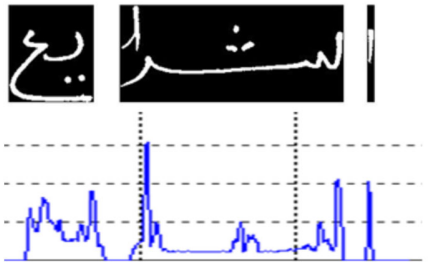


Fig. 2. Sub-words separation

The technique of dividing words into sub-words allows for more accurate baseline detection. In reality, detecting the baseline from a sub-word is more efficient and produces a better outcome.

3.2 Special lines detection

The extraction of special lines ; upper lines (UL), baseline (BL), baseline start (BLS), and lower lines (LL), is an important step in character segmentation, since that the baselines contains the majority of the character connection points.

The uppermost line is UL, and it contains the first black pixel.

$$UL = Argmax_i(Histh(i) > 0) \quad (1)$$

We must find the line with the most black pixels in order to detect the BL.

$$BL = Argmax_i(Histh(i) - Histh(i - 1)) \quad (2)$$

After the BL, the BLS is recognized to have the second highest pixel density.

$$BLS = Argmin_i(Histh(i) - Histh(i - 1)) \quad (3)$$

We identify the lowest line, which contains the first black pixel, starting from the last line; this is the LL.

$$LL = Argmin_i(Histh(i) > 0) \quad (4)$$

By using the equations described, we were able to detect the baselines from the *sub-word obtained (see Fig.3).*

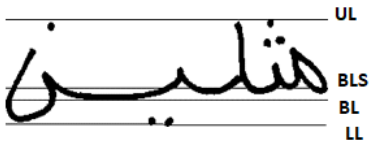


Fig. 3. Special lines detection

The character segmentation algorithm is applied in the next section.

3.3 Preliminary segmentation points detection (PSPD)

We scan the sub-word vertical histogram from right to left to find the value Histv(col) that corresponds to the most frequently occurring values in the column histogram. When such a value is identified, it is referred to as the threshold (Thresh). Because each word has its unique property, a dynamic threshold is quite efficient.

After defining the beginning and end of a character, some rules are applied. The first role (see equation (5)) checks whether the preceding column's horizontal histogram projection is bigger than the threshold. The second one (see equation (6)), on the other hand, determines if the current column's horizontal histogram projection is less than or equal to the threshold. These two conditions guarantee that the character begins in the current $\mathbf{col_i}$ and that the $\mathbf{col_{i-1}}$ is part of the previous character.

$$R_1 : histv(col_i - 1) > Thresh \quad (5)$$
$$R_2 : histv(col_i) <= Thresh \quad (6)$$

The widths of Arabic characters are diverse. Character (ا) denotes the smallest width. To avoid getting non-significant segments we defined **Min_Dist**. When using (equation 7), we make sure that the distance between the current segment's **startCol** and the start of the following segment (**BegNextChar**) is greater than the Min_Dist. Its noteworthy that the widths of the handwritten Arabic characters are long by adding the extension Mata(-) by writers. After multiple experiments, we've fix Min_Dist = **4**.

$$R_3 : |BegNextChar - startCol| > Min\_Dist \quad (7)$$

Some Arabic characters drop below the base line to the point that they become illegible (ن ,و ,ق.). The lower line becomes smaller than or equal to the base line in the final part of these letters. R_4 is used to check if the current column's lower line is bigger than or equal to the baseline.

$$R_4 : LL(col_i) > BaseLine(col_i) \quad (8)$$

Some Others, rise over the basic line. Those characters have a region that is between the base line and the start base line(...,ة,د), but only just few characters (...,ك,ط) go over the start base line. R_5 checks whether the current column's upper line is less than or equal to the baseline start.

$$R_5 : UL(col_i) < BaseLineStart(col_i) \quad (9)$$

R_6 and R_7 ensure that the difference between the lower line and the upper line must be less than or equal to the threshold

$$R_6 : (LL - UL)(col_i) <= Thresh \quad (10)$$
$$R_7 : Histv(coli) <= Thresh \quad (11)$$

The number of vertical transitions (nbVT) is calculated by converting white pixels to black and black pixels to white. Then, using nbVT, R_8 avoid the segmenting isolated characters.

$$R_8 : nbVT(col) == 2 \quad (12)$$

As R_9 verifies, the upper line of the current column must be above the higher line of the starting column to mark the end of the character.

$$R_9 : UL(col_i) <= UL(StartCol_i) \quad (13)$$

Finally, we proposed rule R_{10} ensures that the initial segmentation point is above the base line and below the base line start of the word.

$$R_{10} : BLS(word) < col_i < BL \quad (14)$$

We identified a few segmentation rule failures. The results of detecting preliminary segmentation points of a handwritten word are shown in Fig.4.

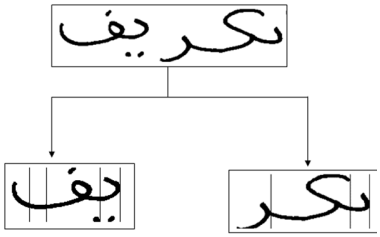


Fig. 4. Example of wrong segmentation

Furthermore, the detected errors are primarily due to the insufficiency of the rules used to delimit some letters with a relatively complicated graphic graphism. For this reason, we merged the fundamental segmentation technique with a classifier decision to attain this target. This classifier is used

to assess the success of the preliminary segmentation point detection step.

3.4 Binary SVM classifier for Segmentation Point Validation (SVM-SPV)

The SVM-SPV algorithm uses a classifier to validate the segmentation points obtained using our basic segmentation algorithm.
To improve results, we propose two roles:

- Under-segmented characters are considered as a single form that will be recognized in its entirety during the recognition stage.
- Over-segmented characters are merged using SVM-SPV (Fig. 5).

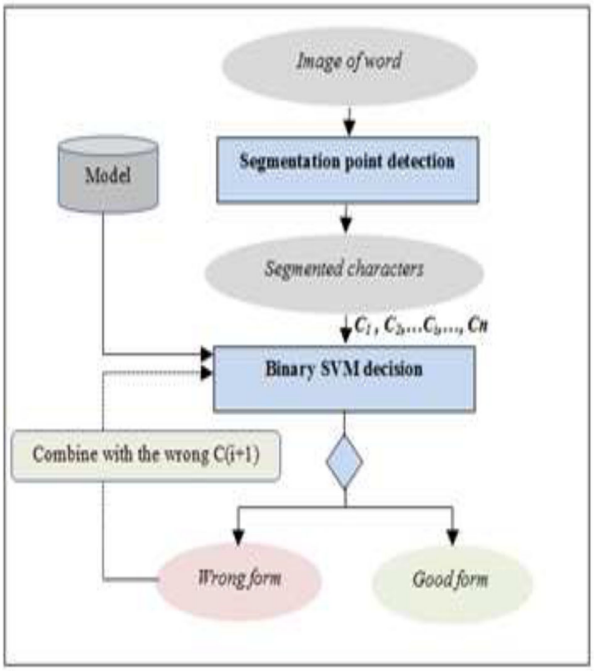


Fig. 5. Binary SVM classifier for Segmentation Point Validation[1]

The motivating result obtained by SVM in the fields of Arabic recognition [15] encourages us to apply it in order to improve our PSPD algorithm. the character (C_i) generated using the basic segmentation algorithm. The initial column sp_1 and the last column sp_2 delimits the character. The character's class is determined based on a binary SVM decision. If C_i is incorrectly classified, the segmentation correction process begins(Fig.6).

```
begin
k := 0                          ▷ k: Maximum number of combinations
i := imageWidth() − 1;
while !( i > 0) do
    if !(isValidSegment(sp1,sp2)) and k < 2 then
        if sp'1 == 0 then   ▷ sp'1:The beginning of the next character
            combine the current segment Ci with the previous one Ci−1 ;
            k++;
        end if
    end if
    k := 0;
    if sp'1 ! = 0)  then
        if !(isValidSegment(Ci)) and k < 2 then
            if !(isValidSegment(Ci+1)) then
                find the next end sp'2 of character Ci by combining Ci and Ci + 1;
            else
                confirm Ci us a good character ;
            end if
        end if
    end if
end while
```

Fig. 6. Segmentation correction process

The classifiers verify the quality of the character C_i. If C_i is a wrong form, we have here two possible scenarios;

1) If the Ci is classified as wrong, it must be logically followed by a wrong form. If the character C_{i+1} is classified as good, we must mark C_i as a good form . The role of this process is to controls the miss-classification risk of the binary decision.
2) If the character C_{i+1} is classified wrong segment, then we must find the next end of the character C_i denoted sp'_2. To obtain a correct segmentation result, we must concatenate C_i with their successor C_{i+1}.

4. Experimental Results

In this study, a variety of experiments were carried out. The performance of the PSPD and SVM-SPV algorithms were compared and the total recognition rate of the system using conventional neuronal network(CNN) was reported.

4.1 Database Description

The dataset IFN/ENIT [19], is consisting of the names of handwritten Tunisian cities. The current version (v1.0p2) is available to researchers. It contains a total of 32492 handwritten city names written by over 1000 writers. The vocabulary is limited to 946 city names. Each city name is made up of one or several words. The IFN/ENIT database is divided into five samples: a, b, c, d and e. Currently, this handwritten base is the most widely used. Numerous researchers have published results on the IFN/ENIT database, which allows us to compare our approach with other works in the field.

During all the experiments, we tested our approach with the "a" sample of the IFN/ENIT database. This set contains 569 samples of city names written by 10 different writers SVM Configuration

4.2 SVM configuration

We organized our data into two categories after acquiring segmented forms PSPD methods.
The good forms are labeled (1), while the bad forms are labeled (2).The classification model will be created using those data. We determined the optimal model by placing it through a series of tests on the training data.The best model is determined by a number of factors. Obtaining the best regularization parameter is the major focus (C) as well as the best kernel parameter (γ). By repeating the classification process for each parameter, the parameters were calculated with 225 combination of(C, γ) where; C = [2^{-2}, ...,2^{12}] and γ = [2^{-10}, ...,2^{4}]. We performed the same tests repeatedly, with four different kernels (linear, polynomial, RBF, and sigmoid). The following table shows a comparison of the best accuracy for each model. We have implemented the our exprimentation using LIBSVM software [20].

Table 1: Predicting the best SVM parameters

Parameters	***KERNELS***			
	Linear	**Polynomial (d=2)**	**RBF**	**Sigmoid**
(C,γ)	***2^3***	(2^{12}; 2^{-7})	(2^{12}; 2^{-6})	(2^{12}; 2^{-6})
Accuracy (%)	96.446	96.65	97.007	96.21

The purpose of SVM is to create a model that predicts the quality of our character. After some testing on the training data set, the best model with the highest accuracy is obtained utilizing RBF kernel ; C=2^{12} and γ =2^{-6}.

4.3 PSPD and SVM-SPV segmentation algorithms performances

The segmentation algorithms performance are evaluated based on three criteria:

- A character that has been separated into more than part components is referred to as " UnderSR ".
- When no segmentation point is located and two characters are segmented into one, an "OverSR " error occurs.
- The term "CorrectSR" refers to a segmentation point that has been extracted appropriately.

Table 2: Results of segmentation evaluation

Algorithms	**Rates(%)**		
	CorrectSR	***UnderSR***	***OverSR***
PSPD	81.32	8.67	10.01
SVM-SPV	89.30	8.67	2

When exmined the results obtained, we found that the over-segmentation was significantly reduced. Despite, the rate of under-segmentation remains significant due to thel overlap present in handwriting writing.
We evaluate the algorithm's performance in addressing the problem of segmenting handwritten words with those acquired by the works in following table.

Table 3: Comparison of our algorithm with literature

Authors	**Rates(%)**	
	CorrectSR	***Dataset***
PSPD(our approach)	81.32	IFN/ENIT
SVM-SPV (our approach)	89.30	IFN/ENIT
[20]	87.9	IFN/ENIT
[21]	82.98	Local dataset

In general, the addition of the SVM classifier's decreased the wrong segmenation results of the fundamental PSPD approach. In fact, misclassified segments were rechecked to ensure that the correct form of characters was obtained. With IFN/ENIT database, our SVM-SPV technique yielded interesting results.

Finally, we can say that our segmentation algorithm has produced results that are close to those seen in the literature. During the recognition phase, more improvement and perfection can be offered in order to achieve the best rates.

4.4 Character classification results

Deep Convolutional Neural Network is a network of artificial neural networks. The design of the neural networks with which we train the parameters in the training phase is known as model architecture. In our optical character recognition system we used LeNet-5 Architecture. This architecture was proposed by Yann LeCun in 1998 [21]. It is suitable for recognition and classification of different classes of objects in small resolution images. The deep architecture consists of 2 convolution layers(1 input layer), 2 pooling layers and 1dense layers(1 output layer). It takes a 28x28 image as input and gives an output the character class.

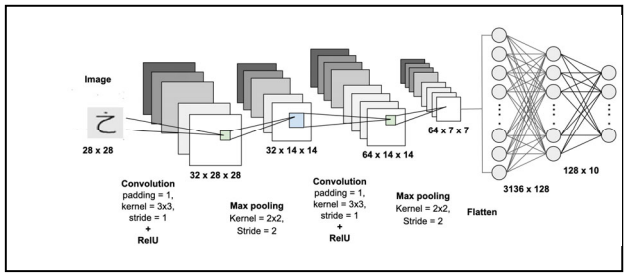


Fig. 7. CNN architecture

In order to assess the recognition rates of handwritten characters, a comparison with other works of the literature is necessary. For this comparison makes more sense, it must be done with systems using the same word base. In this context, our results with the IFN/ENIT database allow us to compare the performance of our system with other work in the following Table.

TABLE I. PREDICTING THE BEST SVM PARAMETERS

Authors	Comparison	
	Recognition rate(%)	*Dataset*
Our approach	78.73	IFN/ENIT
[22]	99.58	IFN/ENIT
[23]	79.60	IFN/ENIT
[24]	54.13	IFN/ENIT

5. Conclusions and Future Work

This paper describes an improved SVM-based segmentation technique for cursive Arabic words. The technique included a knowledge based PSPD Segmentation algorithm. Then an enhanced SVM-SPV algorithm provided better results based on a validation process. Encouraging results were obtained that can increase the overall performance of a segmentation based handwriting recognition system. CNN was used to classify the obtained characters.

In the future, an enhanced CNN architecture will be used to recognize handwritten character for further enhanced performance. The above-mentioned technique will also be tested on a larger dataset to validate the improvements proposed.

Acknowledgments

The authors gratefully acknowledge the approval and the support of this research study by the grant no **SAO-2019-1-10-F-8377** from the deanship of scientific research at northern border university, arar, Kingdom of Saudi Arabia.

References

[1] A. Marwa, K.ZIDI, and K.GHEDIRA. "An efficient and flexible knowledge-based Arabic text segmentation approach." IJCSIS 15.7 (2017).

[2] P.Mohammad Tanvir, and A. Mahmoud. "Offline Arabic handwritten text recognition: a survey." ACM Computing Surveys (CSUR) 45.2 (2013): 1-35.

[3] A.M.Yasser "A survey on Arabic character segmentation." International Journal on Document Analysis and Recognition (IJDAR) 16.2 (2013): 105-126.

[4] J.Lorigo, L. Govindaraju, V. " Segmentation and pre-recognition of Arabic handwriting" In: Proceedings of the 8th International Conference on Document Analysis and Recognition, vol. 2, pp. 605–609

[5] H. A. AlHamadand, R. A. Zitar, "Development of an efficient neural-based segmentation technique for Arabic handwriting recognition" Pattern Recognition, Vol. 43, No. 8, pp. 2773–2798, 2010

[6] A. Alaei, P. Nagabhushan, and U. Pal, "A Baseline Dependent Approach for Persian Handwritten Character Segmentation, " Proc. of ICPR, pp. 1977-1980, 2010

[7] Ghaleb, Hashem, P. Nagabhushan, and Umapada Pal. "Segmentation of offline handwritten Arabic text." 2017 1st International Workshop on Arabic Script Analysis and Recognition (ASAR). IEEE, 2017.

[8] Amara, Marwa, et al. "New rules to enhance the performances of histogram projection for segmenting small-sized Arabic words." International Conference on Hybrid Intelligent Systems. Springer, Cham, 2016.

[9] K.Romeo-Pakker "A new Approach for Latin Arabic Character segmentation 3rd ICDAR." Montreal (1995): 874-877.

[10] Sari, T., Souici, L., & Sellami, M. (2002, August). Off-line handwritten Arabic character segmentation algorithm: ACSA. In Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition (pp. 452-457). IEEE.

[11] Altuwaijri, M.M., Bayoumi, M.A.: A thinning algorithm for Arabic characters usingART2 neural network. IEEETrans. Circ. Syst. II: Analog Digit. Signal Process. 45(2), 260–264 (1998).

[12] Hamid,B Hamid, A.: A neural network approach for the segmentation of handwritten Arabic text. International Symposium on Innovation in Information and Communication Technology. Amman, Jordan (2001).

[13] Hamid, A.,Haraty, R.:Aneuro-heuristic approach for segmenting handwritten Arabic text. ACS/IEEE International Conference on Computer Systems and Applications (AICCSA 2001), pp. 110– 113. Lebanon (2001)

[14] Amara, M., Zidi, K., Ghedira, K., & Zidi, S. (2016, November). New rules to enhance the performances of histogram projection for segmenting small-sized Arabic words. In International Conference on Hybrid Intelligent Systems (pp. 167-176). Springer, Cham.

[15] Amara, M., Ghedira, K, Zidi, K.,Zidi, S.: A Comparative study of multiclass support vector machine methods for Arabic characters

recognition. In: Proceedings of the International Conference on Computer Systems and Applications.(2015)

[16] Amara, M.,Zidi, K., Zidi, S., Ghedira, K.:Arabic Character Recognition Based M-SVM: Review. In: Proceedings of the International Conference on Advanced Machine Learning Technologies and Applications. pp.18-25(2014)

[17] Hamami, L., & Berkani, D. (2002). Recognition system for printed multi-font and multi-size Arabic characters. Arabian Journal for Science and Engineering, 27(1), 57-72.

[18] Zheng, L., Hassin, A. H., & Tang, X. (2004). A new algorithm for machine printed Arabic character segmentation. Pattern Recognition Letters, 25(15), 1723-1729.

[19] Pechwitz, M., Maddouri, S. S., Märgner, V., Ellouze, N., & Amiri, H. (2002, October). IFN/ENIT-database of handwritten Arabic words. In Proc. of CIFED (Vol. 2, pp. 127-136). Citeseer.

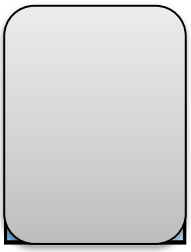
[20] Chang, C. C., Lin, C. J. LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology. 2(3), pp. 27(2011) Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm

[21] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.

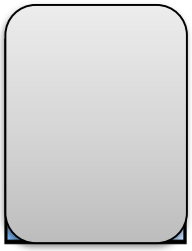
[22] Parvez, M. T., & Mahmoud, S. A. (2013). Arabic handwriting recognition using structural and syntactic pattern attributes. Pattern Recognition, 46(1), 141-154.

[23] Kessentini, Y., Paquet, T., & Hamadou, A. B. (2010). Off-line handwritten word recognition using multi-stream hidden Markov models. Pattern Recognition Letters, 31(1), 60-70.

[24] Elbaati, A., Boubaker, H., Kherallah, M., Ennaji, A., El Abed, H., & Alimi, A. M. (2009, July). Arabic handwriting recognition using restored stroke chronology. In 2009 10th International Conference on Document Analysis and Recognition (pp. 411-415). IEEE.



Marwa Amara was born in Tunisia in 1988. She received her Ph. D degrees in Computer sciences from the National School of Computer Sciences, University of Manouba, Tunisia, in 2016. Currently, she is member of LARIA Laboratory-Tunisia(LA Recherche en Intelligence Artificielle). Her research interests are pattern recognition and artificial intelligence . She is currently an assistant professor at North Border University Saudi Arabia in the Dept. of computer sciences.



Radhia Zaghdoud was born in Tunisia in 1988. She received her Ph. D degrees in Computer sciences from Lille Central School, France, in 2015. Currently, she is member of LARIA Laboratory-Tunisia (LA Recherche en Intelligence Artificielle). Her research interests are optimization Problem and artificial intelligence . She is currently an assistant professor at North Border University Saudi Arabia , where she teach different courses related to computer science.